A Training Method For *VideoPose3D* with Action Recognition

Hao Bai
Zhejiang University - University of Illinois at Urbana-Champaign Institute, ZJUI
Haining, Jiaxing
https://jackgetup.com, Haob.19@intl.zju.edu.cn

Abstract

Action recognition and pose estimation from video are closely related tasks for understanding human motions, but the work to utilize both is rare in literature. In this essay I show an action-recognition-based training method for VideoPose3D, which contributes to the precision of velocity error and speed of training. The model is fed with videos which have exactly the same type of action in the estimation period, and it only requires a small amount of data. Evidence has shown that under a limited amount of epochs and data, our approach outperforms the original research. The code of this project, is available under the open-source MIT License at https://github.com/BiEchi/Pose3dDirectionalTraining.

1. Introduction

Vision-based human motion analysis attempts to understand the movements of the human body using computer vision and machine learning techniques. Technically speaking, the main goal for pose estimation and motion analysis is to rebuild the movements of the locations for human body joints inside videos. Qualitative speaking, lower estimation error means better performance.

The effect between pose estimation and action recognition is mutual. Pose estimation and action recognition can utilize the same dataset [6, 1] Pose estimation can be used for action recognition, with the conclusion that a lower estimation error leads to a more accurate classification of action recognition [19]. Turning around, action priors and also be used to ameliorate the precision of pose estimation [20, 5, 3]. There have also been work combining these two tasks together and perform better in each field [11, 9].

In the field of 3D post estimation by deep learning approach, there are many successful creatures with various types of training methods, like Maximum-Margin Structured training by Li et al. [8], Feedback Loop training by Oberweger et al.[12], and also Semi-Supervised training by Pavllo et al. [13], which we mainly concern with in our work.

VideoPose3D has been very successful because of its high performance and SOTA precision. This work utilizes the datasets **HumanEva** [16] and **Human3.6M** [4]. **Human3.6M** has 11 subjects and 4 viewpoints in total, and 15 types of actions. We've got our motivation that, *VideoPose3D* takes a variety of (instead of a certain type of) actions as training data, but there have been lots of successful cases when using action recognition for pose estimations works well for improving its performance, which gives rise to our project.

2. Related Works and Our Contributions

As mentioned in the Introduction part, our work is based on the previous work *VideoPose3D* [13] and gets its motivation from successful cases to utilize action recognition for pose estimation [20, 5, 3]. In this part we illustrate how these works model and how we can utilize them to perform a better work based on them.

2.1. VideoPose3D

VideoPose3D was the state-of-the-art approach which "utilizes a fully convolutional model based on dilated temporal convolutions over 2D keypoints" [13]. In early researches the main solution was to use recurrent neural networks (RNN) [7], but the research *VideoPose3D* utilized the convolutional neuron network (CNN) to gain an efficient result on the dataset *Human3.6M*, with the inspiration that many authors had mentioned CNN in temporal models.

Technically speaking, *VideoPose3D* takes a series of images as the input of the training network to imitate the functionality as RNN (Figure 1). This ideology has been widely accepted in the current academy and there have been lots of successful cases utilizing this method, e.g. the Spatial-Temporal-CNN used for crowd counting in videos [10], and LSTM-CNN used for face anti-spoofing [18]. From these researches we see the capability of CNN. However, CNN is not a natural temporal network itself, which means it bears a trait of parallelism and lack of temporal memory [15].

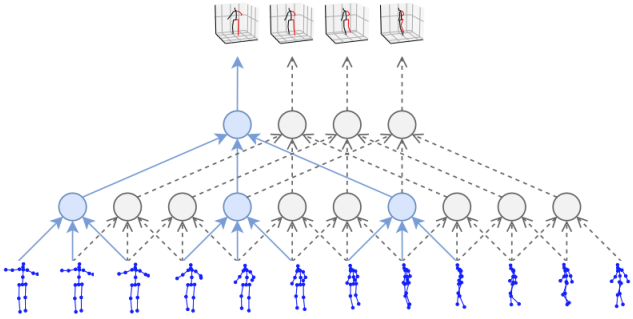


Figure 1: The temporal convilutional model from *VideoPose3D* takes 2D keypoint sequences (bottom) as input and generates 3D pose estimation as out (top).

The overall model of *VideoPose3D* is a semi-supervised learning method (Figure 1).

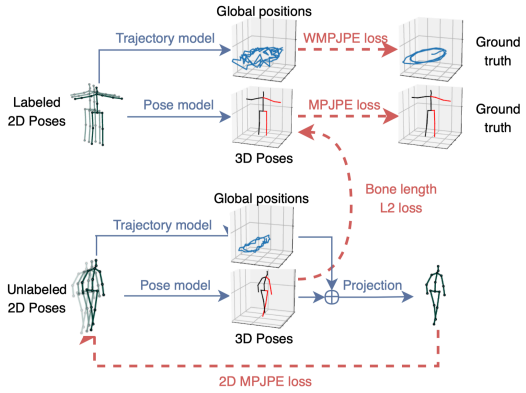


Figure 2: The overall model of *VideoPose3D*

2.2. Action Recognition

The idea of homogenized learning was proposed firstly by Action Estimations [17]. Hitherto, there have been a variety of researches focusing on the combination of action recognition and pose estimation [20]. The basic idea is to begin with 2D appearance-based action recognition based on low-level appearance features. Outputs of the 2D action recognition are used as a prior distribution for the particle-based optimization for 3D pose estimation. Finally, 3D pose-based action recognition is then performed based on pose features extracted from the estimated poses.

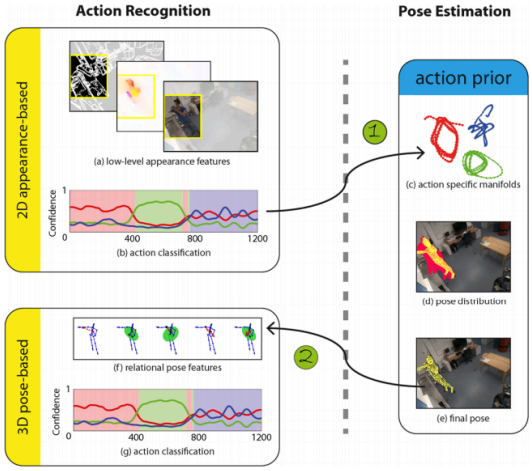


Figure 3: The coupled action recognition and pose estimation framework forwarded by [20].

2.3. Homography Estimation

The idea of using homonegeious actions in a dataset was also inspired from Homography Estimation in some deep images [2].

3. Representation and Modeling

The detailed experimental setup can also be explored in *README.md* in the github repo. To reproduce the results, you need to test for 15 times for each type of action, and it's preferable to test for multiple times. It's more convenient to use *bash* scripts to save time.

Our work focuses on using certain types of actions for training data. For example, say there are 4 actions for all subjects, i.e. sitting, walking, discussing, and posing. The original *VideoPose3D* model takes all 4 types of data as training data (Figure 4, left), and they estimate results for all action in the end. Our model, instead, takes only one action as training data (Figure 4, right), and we estimate ONLY result for this certain action in the end. Absolutely, this process will iterate through all actions if all of them need to be estimated.

Compared to the original model we utilized a different data-processing approach. The original approach mentioned in *VideoPose3D* uses all actions in training data, and the estimating effect was good overall. We utilize the method that we only take one type of action for training, and the total amount for training should be the same. Mathematically, take the action *Sitting* for example, it can be expressed as*

$$\sum_{i \in actions} f(i) = f(Sitting) \cdot t \qquad (1)$$

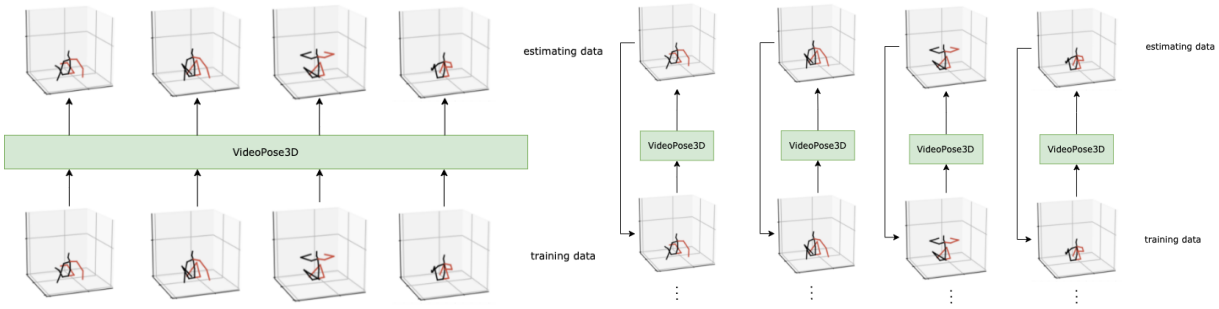


Figure 4: The comparison of the original model (left) and our model (right). The original model takes all types of actions for training, but we only take a certain type of data as training data, as an approach to utilize action recognition for pose estimation.

where f stands for the frames of subject, and t means the number of epochs of our action-based training. Note that the unit of t is unit-epoch instead of epoch. The unit-epoch stands for 1 epoch of one action. According to the dataset we utilize, *Human3.6M*, there are 15 actions for each subject. Thus, the relationship between epoch and unit-epoch can be expressed mathematically as below.

$$t_0 = t_{unit} \cdot 15 \tag{2}$$

In this approach, we utilize only one time of action, which means we save more data; and we also produce the results in the same time, the evidence can be illustrated by formula (1). After testing, we conclude that our work outperforms the original work by 11%.

The principle of our work is shown below.

In our model we also changed the process of training in the original research. In the original research, it takes 15 unit-epochs to run a single epoch, because there are 15 types of actions. In our work, we can tweak the number of unit-epochs not necessarily to be a multiple of 15, which makes it more flexible to observe and iterate. Shown below is the algorithm for our iterative training when not so many epochs are available (we take 2 epochs for example).

Algorithm 1: Original *VideoPose3D* Model

Data: Training data, testing data, ground-truth data, total training epoch = 2.
Result: Error value.

```
for EPOCH in 2 do
    ERR=VideoPose3D(TRAIN, TEST, GT);
    print(ERR);
end
return ERR;
```

Algorithm 2: Our Action-based Approach

Data: Training data, testing data, ground-truth data, total training epoch = 2, action type.
Result: Error value.

```
for EPOCH in 2 · 15 do
    ERR FOR ACTION=VideoPose3D(TRAIN, TEST, GT, ACTION);
    print(ERR FOR ACTION);
end
return ERR FOR ACTION;
```

4. Experiments

Shown below are the results of the experiment. Expect a minor error when testing on your own. Note that all unit are millimeter.

4.1. Argument-based Training Results

In this part we estimate the errors with different combinations of arguments for model training. The epoch of the original test was set to be 15 unit epochs (i.e. 1 epoch), and the epoch of the action-based test was set to 1 epoch. In the table below (Table 1 & Table 2), we illustrate the pose estimation results with respect to different types of model with different training arguments (i.e. the number of receptive field Frames **F**, the number of Unit Epochs **UE**), 15 different actions, and two different protocols (i.e. MPJPE and Velocity Error of MPJPE).

In the tables the model with the best performance is in bold, and we set the argument as the highest classification standard for better comparison. The visualized Argument part is labeled in the corresponding letter of Figure 5.

Arguments	Model	Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Smoke	Wait	WkD.	Walk	WkT.	**Avg**
	VideoPose3D	67.0	**65.8**	68.7	72.5	**74.2**	**87.2**	**65.5**	**73.1**	85.6	117	73.6	**70.8**	81.1	63.5	67.3	75.5
	Ours	**61.9**	69.2	**62.0**	**68.4**	75.7	88.2	74.5	76.9	**81.5**	**97.9**	**71.1**	80.9	**80.7**	**49.9**	**59.2**	**73.2**
	VideoPose3D	**54.5**	**61.4**	56.3	**58.6**	**61.3**	**68.6**	**57.6**	**60.6**	70.5	84.7	60.5	**59.1**	68.2	51.8	**53.1**	**61.8**
	Ours	70.5	71.9	**52.1**	61.4	69.0	82.6	70.6	82.1	**70.4**	**79.4**	**60.3**	79.5	**56.1**	**44.2**	56.2	67.1
	Pavlakos [18]	67.4	71.9	66.7	69.1	72.0	77.0	65.0	68.3	83.7	96.5	71.7	65.8	74.9	59.1	63.2	71.9
	Luvizon [8]	49.2	51.6	47.6	50.5	51.8	60.3	48.5	51.7	61.5	70.9	53.7	48.9	57.9	44.4	48.9	53.2
	VideoPose3D	**46.6**	**47.4**	45.2	**46.2**	**49.0**	**56.7**	**46.4**	**47.2**	**59.9**	68.2	**48.1**	**46.2**	49.4	32.9	**34.3**	**48.2**
	Ours	47.9	48.8	**45.1**	48.4	51.7	62.8	47.1	59.4	61.2	**64.7**	48.2	59.3	**46.6**	**31.4**	35.5	50.5

Table 1: Protocol 1, MPJPE Error

Arguments	Model	Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Smoke	Wait	WkD.	Walk	WkT.	**Avg**
	VideoPose3D	**10.7**	**12.2**	10.2	12.4	11.1	**11.0**	10.5	12.3	**11.4**	**13.8**	11.0	**10.6**	12.7	12.9	12.4	11.7
	Ours	10.9	13.0	**9.59**	**11.8**	**10.3**	11.4	**10.1**	**10.5**	12.2	14.6	**10.9**	11.5	**12.5**	**12.6**	**12.2**	**11.6**
	VideoPose3D	3.84	4.02	3.14	4.42	3.31	3.58	**3.44**	3.93	3.14	4.21	3.36	**3.34**	4.68	4.51	4.08	3.80
	Ours	**3.78**	**4.28**	**3.01**	**4.27**	**3.14**	**3.50**	3.79	**3.65**	**2.84**	**3.90**	**3.05**	3.55	**4.37**	**3.77**	**3.49**	**3.63**
	VideoPose3D	**3.01**	3.21	**2.33**	**3.55**	2.33	**2.83**	**2.77**	3.23	2.11	3.01	2.44	**2.43**	3.82	3.33	**2.86**	**2.49**
	Ours	3.14	**3.20**	2.48	3.62	**2.31**	2.91	3.04	**3.11**	**2.07**	**2.88**	**2.23**	2.73	**3.66**	**3.04**	2.97	3.08

Table 2: Protocol 2, MPJPE Velocity Error

4.2. Temporal-related Training Results

The time for training data is a crucial standard for deep learning, like the success of invention of Faster R-CNN [14]. In this part we take a deep insight of the time consumption and precision-time rate between the original model and our Eating-based model (left), and the relationship between our work and the original work using protocol MPJPE with the lapse of epochs (right) in Table 4. The corresponding visualization is shown in figure 6.

In the second subtable the precision-time rate θ is calculated by the formula below, where ϵ_0 stands for the regulated error when training to the half of the process, ϵ_t represents the error after a long time of training (in our case 80 epochs), $i \in \{0, 1\}$, $k = 1.2$ and t stands for the real time consumed for each training.

$$\epsilon_0^{(1,2)} = \frac{\epsilon_{\lfloor \frac{t}{2} \rfloor}^{(1)} + \epsilon_{\lfloor \frac{t}{2} \rfloor}^{(2)}}{2} \cdot k \tag{3}$$

$$\theta^{(i)} = \frac{\epsilon_0^{(1,2)} - \epsilon_t^{(i)}}{t} \tag{4}$$

Object	VP3D-Avg	Ours-Avg
TC (40 E)	84972 sec.	28848 sec.
MPJPE	50.27 mm	50.49 mm
Velocity-M	2.66 mm	3.02 mm
TC (80 E)	176976 sec.	56921 sec.
MPJPE	48.22 mm	50.54 mm
Velocity-M	2.49 mm	3.08 mm
MPJPE ϵ_0	60.46	
Velo-M ϵ_0	3.408	
MPJPE TPR	$6.92 \cdot 10^{-5}$	$17.4 \cdot 10^{-5}$
Velo-M TPR	$5.19 \cdot 10^{-6}$	$5.76 \cdot 10^{-6}$

Table 3: The property cluster with receptive field = 243 frames and training epochs = 1200 unit epochs.

Epochs	VP3D-Avg	Ours-Avg	VP3D-Eat	Ours-Eat
1	59.11	57.52	54.22	51.14
2	56.26	52.06	52.21	48.72
3	61.73	51.28	51.17	47.14
4	54.24	51.41	49.22	45.28
5	51.17	51.11	49.74	45.17
6	50.16	50.11	51.12	46.28
7	50.28	50.37	50.71	45.82
8	50.19	50.25	50.23	47.14
9	50.24	50.34	49.17	47.22
10	50.14	50.94	49.82	46.32
30	49.56	50.56	47.71	45.64
40	50.27	50.49	46.14	45.18
50	48.17	50.29	45.61	45.12
80	48.22	50.54	45.22	45.14

Table 4: The comparison between our work and the original work under receptive field = 243 frames and training epochs = 1200 unit epochs.

5. Conclusion

Concluding all results from the Experiment part we make conclusions that our model can be used to solve action-oriented problems, pick any kind of actions you want to train and estimate, accelerate the learning process, and save your data.

5.1. A Lightweight Solution For Action-oriented Pose-estimation Problems.

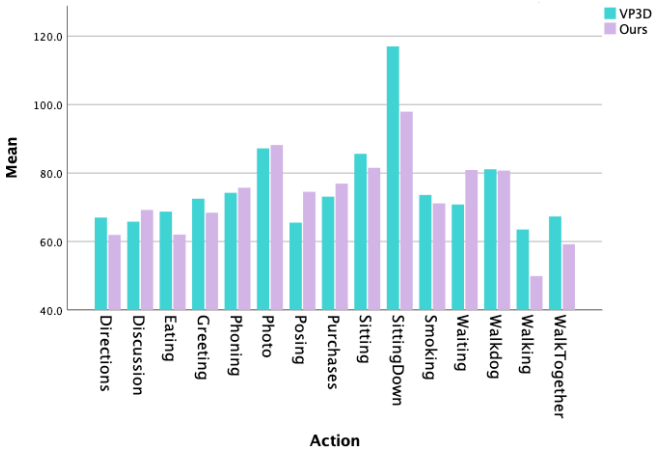
Our model can serve as a viable solution for action-oriented problems. If there is a request that only a certain action (e.g. Eating) is needed to be trained and estimated, we can use our method to act as an alternative other than *VideoPose3D*. Evidence has proved that our model has a similar pose estimation ability with *VideoPose3D* using MPJPE Protocol (Table 1) and a better performance than *VideoPose3D* under a middle-size reception field and short training epoch using Velocity Error of MPJPE Protocol (Table 2). Also, our model converges much quicker than the original model (Table 4-left and Figure 6) and uses less data (as mentioned in Part 3).

These characteristics imply that our model is a good candidate for tasks where a specific action is needed to be trained and estimated, and the learning cost is limited. Our model doesn't require a large amount of computational resources.

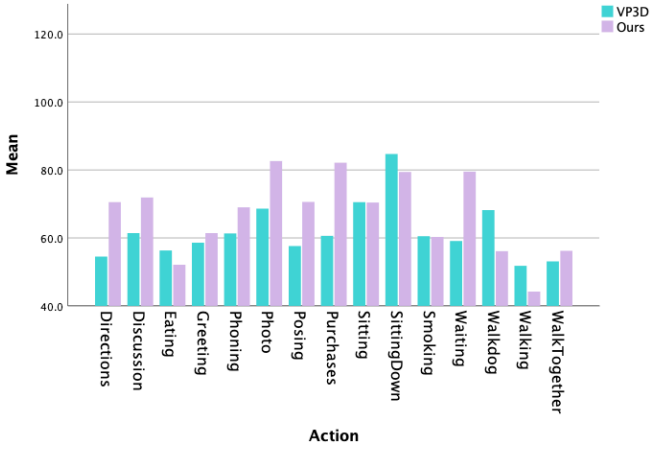
5.2. A Fast Solution For Common Pose-estimation Problems.

Our model can also be used for a common pose-estimation problem. Although its best-performance is not as good as *VideoPose3D* (Table 1 and 2), it requires a much shorter time for training and gives a higher precision-time rate (Table 4-left). The flexibility that it can be trained using different actions can also serve as a way to improve the performance of a certain kind of action when the result is not yet satisfactory.

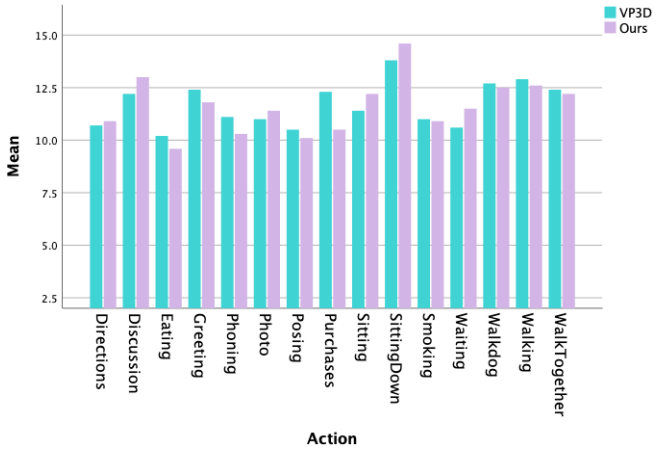
5.3. Appendix



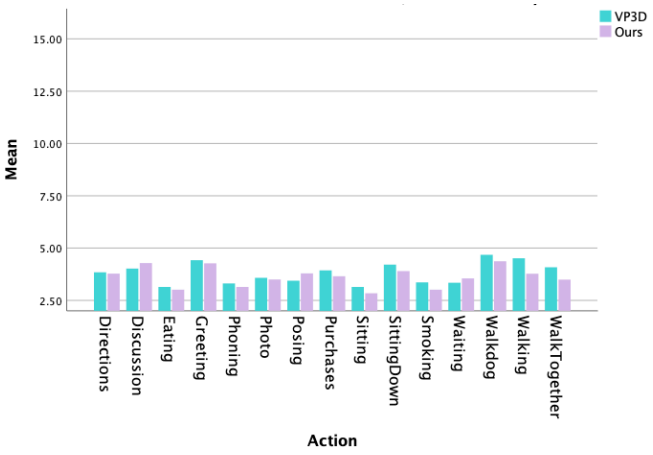
(a) 1F, 15UE, MPJPE



(b) 27F, 15UE, MPJPE



(c) 1F, 15UE, Velocity Error of MPJPE



(d) **27F, 15UE, Velocity Error of MPJPE**

Figure 5: Error Visualization for experimental results

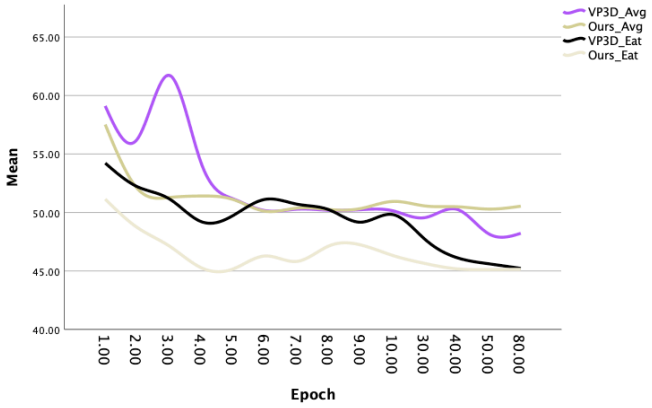


Figure 6: The temporal results for the two models (*VideoPose3D* and our model) and two standards (average and only eating).

References

[1] Mykhaylo Andriluka, Umar Iqbal, Eldar Insafutdinov, Leonid Pishchulin, Anton Milan, Juergen Gall, and Bernt Schiele. Posetrack: A benchmark for human pose estimation and tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5167–5176, 2018.

[2] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Deep image homography estimation. *arXiv preprint arXiv:1606.03798*, 2016.

[3] Juergen Gall, Angela Yao, and Luc Van Gool. 2d action recognition serves 3d human pose estimation. In *European Conference on Computer Vision*, pages 425–438. Springer, 2010.

[4] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1325–1339, 2013.

[5] Umar Iqbal, Martin Garbade, and Juergen Gall. Pose for action-action for pose. In *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, pages 438–445. IEEE, 2017.

[6] Hueihan Jhuang, Juergen Gall, Silvia Zuffi, Cordelia Schmid, and Michael J Black. Towards understanding action recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 3192–3199, 2013.

[7] Kyoungoh Lee, Inwoong Lee, and Sanghoon Lee. Propagating lstm: 3d pose estimation based on joint interdependency. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 119–135, 2018.

[8] Sijin Li, Weichen Zhang, and Antoni B Chan. Maximum-margin structured learning with deep networks for 3d human pose estimation. In *Proceedings of the IEEE international conference on computer vision*, pages 2848–2856, 2015.

[9] Diogo C Luvizon, David Picard, and Hedi Tabia. 2d/3d pose estimation and action recognition using multitask deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5137–5146, 2018.

[10] Yunqi Miao, Jungong Han, Yongsheng Gao, and Baochang Zhang. St-cnn: Spatial-temporal convolutional neural network for crowd counting in videos. *Pattern Recognition Letters*, 125:113–118, 2019.

[11] Bruce Xiaohan Nie, Caiming Xiong, and Song-Chun Zhu. Joint action recognition and pose estimation from video. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1293–1301, 2015.

[12] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit. Training a feedback loop for hand pose estimation. In *Proceedings of the IEEE international conference on computer vision*, pages 3316–3324, 2015.

[13] Dario Pavllo, Christoph Feichtenhofer, David Grangier, and Michael Auli. 3d human pose estimation in video with temporal convolutions and semi-supervised training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7753–7762, 2019.

[14] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99, 2015.

[15] Hoo-Chang Shin, Holger R Roth, Mingchen Gao, Le Lu, Ziyue Xu, Isabella Nogues, Jianhua Yao, Daniel Mollura, and Ronald M Summers. Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning. *IEEE transactions on medical imaging*, 35(5):1285–1298, 2016.

[16] Leonid Sigal, Alexandru O Balan, and Michael J Black. Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International journal of computer vision*, 87(1-2):4, 2010.

[17] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *Proceedings of the IEEE international conference on computer vision*, pages 3551–3558, 2013.

[18] Zhenqi Xu, Shan Li, and Weihong Deng. Learning temporal features using lstm-cnn architecture for face anti-spoofing. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 141–145. IEEE, 2015.

[19] Angela Yao, Juergen Gall, Gabriele Fanelli, and Luc Van Gool. Does human action recognition benefit from pose estimation?". In *Proceedings of the 22nd British machine vision conference-BMVC 2011*. BMV press, 2011.

[20] Angela Yao, Juergen Gall, and Luc Van Gool. Coupled action recognition and pose estimation from multiple views. *International journal of computer vision*, 100(1):16–37, 2012.